Forecasting Hourly Ambulance Demand for Oslo, Norway: A Neuro-Symbolic Method

Erling Van De Weijer[0009-0007-5932-4295], Odd André Owren[0009-0006-3784-0507], and Ole Jakob Mengshoel[0000-0003-2666-5310]

Norwegian University of Science and Technology
erling.weijer@gmail.com, oddandreowren@gmail.com, ole.j.mengshoel@ntnu.no
https://www.ntnu.edu/employees/ole.j.mengshoel

Abstract. Forecasting ambulance demand is critical for emergency medical services to allocate their resources as efficiently as possible. This work uses data from Norway's Oslo University Hospital (OUH) to forecast hourly ambulance demand in Oslo and Akershus. To forecast demand, we developed a neuro-symbolic method, DeANN. DeANN integrates statistical decomposition and artificial neural network methods. Statistical decomposition computes trend, seasonal, and residual components from the ambulance demand time series. Using these components, we apply a multilayer perceptron and regression to compute an overall ambulance demand forecast. Based on experimental results, we conclude that our proposed neuro-symbolic approach for ambulance demand forecasting outperforms several baseline models. Our best neuro-symbolic model has a mean squared error of 21.68 and improves on previous results for the OUH data set.

Keywords: Ambulance demand forecasting · Machine learning · Artificial neural networks · Statistical decomposition.

1 Introduction

Context. Emergency medical services (EMSs) respond to emergency calls and provide pre-hospital care and transport. After a medical incident occurs, an EMS operator is typically notified. The operator assesses the situation and available resources while also providing instructions to the caller before deciding which resource, typically an ambulance, to dispatch. When located by the ambulance, the patient will receive medical care at the scene. If necessary, the ambulance then transports the patient to a medical facility while care is provided.
Challenges. A fleet of ambulances is generally on duty to minimize EMS response times. Discussing fleet management, Soeffker *et al.* identify three high-level goals: transportation of goods, transportation of people, and delivery of services [13]. Emergency vehicles including ambulances generally seek to achieve two of these three goals, namely transportation of people and delivery of services. To achieve these goals, forecasting medical incidents is considered a key component of a comprehensive EMS system [9]. Due to medical incidents being

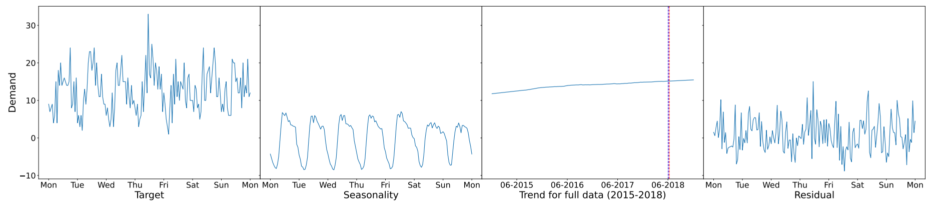

Fig. 1: A medical incident time series for a week in June 2018 is shown in the leftmost panel. The three remaining panels illustrate the STL decomposition. The mid-left panel shows the seasonal component. The mid-right panel is the full trend for our data, with blue and red vertical lines to mark the week of the other panels. The rightmost panel is the resulting residual.

uncertain, dynamic, and time-critical [12,5,10], forecasting and managing such incidents is among the most difficult fleet management problems. These difficulties are partly reflected in Figure 1's leftmost panel, which shows the demand for ambulance services (y-axis) for an arbitrary week in June 2018 (day of the week on the x-axis) for Oslo, Norway.

Many authors have compared, for time series forecasting, the performance of statistical and machine learning (ML) methods [8,7]. These methods have varying strengths and weaknesses when it comes to time series forecasting. Statistical decomposition methods (see [2,15,1,16]), such as the seasonal-trend decomposition procedure based on loess (STL) [2], can capture components in a time series. Often, these components are interpretable to humans, see Figure 1. However, the components may only capture coarse temporal patterns in data, and thus forecasting accuracy may suffer. In contrast, ML methods such as artificial neural networks (ANNs) may perform well in terms of accuracy metrics like mean squared error (MSE) or mean absolute error (MAE). However, interpretability may suffer, since ANNs typically are high-dimensional models with complex topologies and thousands if not millions of numeric parameters.

Contributions. In this paper,[1] by combining statistical decomposition and ANNs for the purpose of time series forecasting, we hope to take the best of both types of methods while overcoming or at least reducing their weaknesses. Instead of asking the question "which, among a set of statistical and ML methods, is best?" we consider the question "how can we best combine such methods?". Our answer is DeANN, a multi-scale decomposition method hybridized with ANNs. For our ambulance demand data set this neuro-symbolic method gives better accuracy than benchmark methods while also providing an interpretable time series decomposition. Figure 1 shows an example of STL decomposition.

2 Background and Related Work

Data Set. The OUH emergency medical care department and the Norwegian National Advisory Unit for Prehospital Emergency Medicine (NAKOS) provided

[1] This paper builds upon the MS thesis of Van De Weijer and Owren [14].

our data set. The data set contains an anonymized set of incidents from the 1st of January, 2015, to the 11th of February, 2019. Anonymization is achieved by mapping each incident to a 1x1km grid cell, acquired from Statistics Norway (SSB), and assigning the identifier (ID) of the cell to the incident. The anonymization allows us to maintain a high resolution of the data while still preserving the anonymity of those involved in the incidents.

The initial processing of the data set consists of three steps: removal of duplicate rows, filtering events outside the years 2015-2019, and removal of events outside the borders of Oslo and Akershus.[2] There is a stable increase in incidents each year, except for 2019 where the data set ends in February. Most incidents are acute or urgent, making up 80.9% of total incidents in the data set. Similar to earlier work [5] we use a filtered data set with events until the end of the year 2018 in experiments. This data set is split into training, validation, and test sets. We use the first 80% of the data for training, the following 10% for validation, and the final 10% for testing.[3]

Time Series Decomposition and Forecasting. A nonstationary time series may be decomposed into a seasonal component, trend component, and residual. Seasonal-trend decomposition using local regression (STL) has been combined with deep learning to forecast tourism demand [16]. Basak *et al.*'s HyperSTL method [1] uses STL decomposition to preserve extrema in time series smoothing in soil moisture analysis. The result is a smooth trend component replicating the time series while also providing forecasts without applying ANNs.

Another decomposition method is singular spectrum analysis (SSA) [15]. SSA is a non-parametric spectral estimation method that can be used to decompose a time series into a sum of components. Unlike STL, SSA does not require the period length to be predetermined. SSA has successfully been applied for decomposition of rainfall times series [15].

Ambulance Demand Forecasting. Our focus is emergency call volume prediction, or demand forecasting, for a certain geographical area. For such an area, the MEDIC method used in industry [12] predicts demand for a given hour of the day h, day of the week d, week w, and year y:

$$\hat{y}_{h,d,w,y} = \frac{\sum_{i=0}^{4}\sum_{j=1}^{4} y_{h,d,w-j,y-i}}{20}. \tag{1}$$

Intuitively, MEDIC's predicted value $\hat{y}_{h,d,w,y}$ for a given time on a given weekday is based on the value at the same time of day on the same weekday for the four previous weeks for the past five years.

Setzler *et al.* [12] design an ANN to forecast ambulance demand in Mecklenburg County, North Carolina, USA, for several combinations of 1-3 hour time buckets and 2-4 square mile grids. Their results are compared to the MEDIC method and the proposed model scored slightly better on a 4×4-mile grid. Zhou *et al.* [17] propose spatio-temporal kernel density estimation (stKDE) to address

[2] The data cannot be made publicly available due to privacy concerns. After preprocessing the data set we get approximately 560 000 medical incidents [14].

[3] Details about the spilt are discussed on pages 36–37 in the Master's thesis [14].

the challenges of ambulance demand forecasting. The authors model Toronto's ambulance demand over a continuous spatial domain and a temporal domain of one-hour intervals. The resulting model scored highly on forecast accuracy while requiring low computational power.

Different ML methods for predicting hourly ambulance demand in the Oslo and Akershus region have been studied [5]. Models for forecasting hourly total demand (volume models) and hourly demand per 1 × 1 km grid cell (complete models) have both been considered. These forecasting models include both multi-layer perceptron (MLP) and long short-term memory (LSTM) using different combinations of hidden layers and nodes in each layer. Volume models include both MLP and LSTM as well as the average demand distribution for the entire test data. The best-performing model, surpassing the performance of both MEDIC and Setzler's model [12], is an MLP using hour, day of the week, and month as input features [5].

3 Forecasting Medical Incidents: The DeANN Method

We propose DeANN, a neuro-symbolic forecasting method, adapting concepts from the work of Zhang *et al.* [16] to a new domain. DeANN performs a multi-scale time decomposition and hybridizes with ANNs, specifically MLPs. The MLP approach is inspired by Huang *et al.* [6], using a genetic algorithm (GA) for weight initialization.

3.1 Preprocessing and Data Analysis

We aggregate OUH medical incident data for the Oslo area into a single time series of total hourly demand as part of our preprocessing. Exploratory data analysis indicated seasonality and trend in the data; this led us to apply the STL and SSA methods to our data to decompose the time series to extract temporal patterns

3.2 Symbolic Decomposition Methods

Our goal with decomposition is to extract interpretable information from the time series before MLP training. We study the STL and SSA decomposition methods. When comparing these methods and their hyperparameter settings, we use the same error metrics as we use for comparing models as the decomposition with the lowest error will also extract the most information from our time series.

STL Decomposition and ANNs. STL extracts seasonality and trend from our time series so that we can train our MLP on the residual. We use an additive version of STL, such that the sum of each component returned from the decomposition results in the original time series. When using STL for decomposition of a time series $\boldsymbol{y}$, via STL($\boldsymbol{y}$, p, $\boldsymbol{\theta}$), a predefined period p must be set. In Figure 2b we use STL($\boldsymbol{y}$, p, $\boldsymbol{\theta}$) to compute an STL decomposition with period p (measured in days) of a univariate time series $\boldsymbol{y} = (y_1, \ldots, y_t, \ldots)$, where y_t represents the

a DeANN Neuro-Symbolic Method

```
Input: Time series $\boldsymbol{y}$, input features $\boldsymbol{x}$, horizon $h$
Parameter: STL parameters $\boldsymbol{\theta}_s$, $\boldsymbol{\theta}_t$, ANN initialization parameters $\boldsymbol{\theta}_{\text{ANN}}$
Output: Forecast $\hat{y}_{t+h}$
1: Let $(\boldsymbol{s}_s, \boldsymbol{t}_t, \boldsymbol{y}'') = \text{TwiceSTL}(\boldsymbol{y}, \boldsymbol{\theta}_s, \boldsymbol{\theta}_t)$
2: Let $\hat{s}_{t+h} = \text{Extend}(\boldsymbol{s}_s)$
3: Let $\hat{t}_{t+h} = \text{Regression}(\boldsymbol{t}_t)$
4: Let $\mathcal{N} = \text{CreateANN}(\boldsymbol{\theta}_{\text{ANN}})$
5: $\mathcal{N}.\text{train}(\boldsymbol{x}, \boldsymbol{y}'')$
6: Let $\hat{y}''_{t+h} = \mathcal{N}.\text{predict}(x_{t+h})$
7: Let $\hat{y}_{t+h} = \hat{y}''_{t+h} + \hat{s}_{t+h} + \hat{t}_{t+h}$
8: return $\hat{y}_{t+h}$
```

b TwiceSTL

```
Input: Time series $\boldsymbol{y}$
Parameter: STL parameters $\boldsymbol{\theta}_s$, $\boldsymbol{\theta}_t$
Output: Seasonality $\boldsymbol{s}_s$, trend $\boldsymbol{t}_t$, residual $\boldsymbol{y}''$
1: Let $(\boldsymbol{s}_s, \boldsymbol{t}_s, \boldsymbol{r}_s) = \text{STL}(\boldsymbol{y}, p_s, \boldsymbol{\theta}_s)$, where $p_s \in \boldsymbol{\theta}_s$
2: Let $\boldsymbol{y}' = \boldsymbol{t}_s + \boldsymbol{r}_s$.
3: Let $(\boldsymbol{s}_t, \boldsymbol{t}_t, \boldsymbol{r}_t) = \text{STL}(\boldsymbol{y}', p_t, \boldsymbol{\theta}_t)$, where $p_t \in \boldsymbol{\theta}_t$
4: Let $\boldsymbol{y}'' = \boldsymbol{s}_t + \boldsymbol{r}_t$
5: return $(\boldsymbol{s}_s, \boldsymbol{t}_t, \boldsymbol{y}'')$
```

Fig. 2: Pseudo-code for the STL-based (right, in 2b) version of our DeANN neuro-symbolic method (left, in 2a).

number of incidents at time t.[4] Resulting from the STL decomposition with period p and parameters $\boldsymbol{\theta}$ are $\boldsymbol{s} = (s_1, \ldots)$, $\boldsymbol{t} = (t_1, \ldots)$, and $\boldsymbol{r} = (r_1, \ldots)$, which respectively are seasonality, trend, and residual time series.

We tested decompositions at different periods, such as daily, weekly, and yearly. Initially, we used the trend from a single weekly STL decomposition. However, we later found that doing a 2-step STL decomposition where we attain the seasonality of period p_s days in the first step and trend of period p_t days in the second step gave the most reasonable trend and residual for further forecasting. This resulted in the STL calls in lines 1 and 3 in Figure 2b for TwiceSTL.

Our overall neuro-symbolic DeANN method is shown in Figure 2. To make a forecast from a decomposed time series for a time horizon h, DeANN (Figure 2a) approximates the trend component $\boldsymbol{t}_t$ by polynomial regression to create a forecast value $\hat{t}_{t+h}$. An ANN trained on the residual component $\boldsymbol{y}''$ computes $\hat{y}''_{t+h}$. The final forecast value $\hat{y}_{t+h}$ is the sum of the forecast values $\hat{t}_{t+h}$ and $\hat{y}''_{t+h}$ as well as the extracted seasonal component $\hat{s}_{t+h}$. To avoid overfitting, we prefer a lower degree of polynomial regression. (Alternatively, we could keep the trend component and use the ANN to learn this feature from data.)

SSA Decomposition and ANNs. SSA is another statistical method for decomposition [15]. Unlike STL, SSA does not require a predetermined period before decomposition. No assumptions about the length of the optimal period are needed. Thus, our DeANN method for using SSA is slightly different than our STL-based DeANN method presented in Figure 2.

Due to limited space we only present the main points of our SSA-based method here. A key parameter in SSA is the grouping used for reconstruction.

[4] To simplify the pseudo-code, we say $p \in \boldsymbol{\theta}$ to indicate that p is among STL's input parameters $\boldsymbol{\theta}$, despite these not being a set but a tuple.

The grouping decides which components from the deconstructed time series to be used for reconstruction. We study two grouping methods using either a periodgram or a W-correlation matrix. When grouping using periodgram, one looks at the contribution provided by each component and includes components where the contribution to the reconstruction is greater than a specified threshold. The other option uses hierarchical clustering according to SSA algorithm 2.15 (see [4]). The W-correlation matrix is used as proximity matrix. Either way, the resulting decomposition consists of two reconstructed series, one containing the decomposed trend and seasonality, and one for the residual.

To summarize, SSA decomposes our time series $\boldsymbol{y}$ into $\boldsymbol{g}$ and $\boldsymbol{r}$ such that $\boldsymbol{y} = \boldsymbol{g} + \boldsymbol{r}$. Here, $\boldsymbol{g}$ is the grouping of components found in SSA-decomposition (representing seasonalities and trends) while $\boldsymbol{r}$ is the residue (a noise term). The residue $\boldsymbol{r}$ is what is "left over" after SSA decomposition. SSA forecasting methods [3] compute forecasts $\hat{g}_{t+h}$ given a time horizon h. We train an MLP on the residual component $\boldsymbol{r}$ to create predictions $\hat{r}_{t+h}$. We get our final forecast $\hat{y}_{t+h}$ by summing these components, such that $\hat{y}_{t+h} = \hat{g}_{t+h} + \hat{r}_{t+h}$.

3.3 Neural ML Methods

Our design of the ANN $\mathcal{N}$, see lines 4, 5, and 6 in Figure 2a, is inspired by an existing MLP architecture with state-of-the-art performance on this data [5]. However, we use linear activation functions instead of ReLU (see [5]) in models that forecast using decomposed data, to have the ability to output negative values without additional weights on the output layer. Usually, this is not an issue in demand forecasting. However, after seasonality and trend are extracted by Figure 2b, the residual will have a mean close to 0, with a deviation allowing for negative values. This change is only done to the final layer of our model to keep the architecture as close as possible to the original. Our method differs from previous work [5], as we incorporate a symbolic decomposition step TwiceSTL (Figure 2b). We are adjusting the activation functions of our model accordingly.

Feature Selection with MLP We now consider the features input to the algorithm in Figure 2a, which impact $\mathcal{N}$'s architecture and performance (see lines 4, 5, and 6 in Figure 2a). Our data analysis shows that ambulance demand increased annually, from 118 384 incidents in 2015 to 146 416 incidents in 2018 [14].[5] Thus, including the year among the input features $\boldsymbol{x}$ to our ANNs may improve the previous model [5]. As the trend is almost linear, we experiment with including the year both as a one-hot encoded vector Y_O and as a numeric value Y_N with $0 \leq Y_N \leq 1$, see Table 1. The year comes in addition to one-hot-encoded features already included: hour of the day (H), day of the week (D), and month (M).

[5] One reason for this increase is Oslo's increasing population during the time period studied. A hypothetical decrease in demand should be picked up by STL's trend component if the decrease is significant enough.

Since 2015 is the first year present in our preprocessed data set, to get Y_N we first subtract 2015 from the year, thus the year is a value between 0 and 4. We then divide by 4, such that the final representation is a number $0 \leq Y_N \leq 1$.[6]

Table 1: Alternative representations for year Y as an ANN feature: one-hot encoded vector Y_O or numeric value Y_N.

Year	$\boldsymbol{Y_O}$	$\boldsymbol{Y_N}$
2015	[1,0,0,0,0]	0
2016	[0,1,0,0,0]	0.25
2017	[0,0,1,0,0]	0.5
2018	[0,0,0,1,0]	0.75
2019	[0,0,0,0,1]	1

Our experiment compares three different input features in an MLP model with the same architecture but different input layers. The different features sets $\boldsymbol{x}$ are thus *HDM*, $HDMY_O$, and $HDMY_N$.[7] The $HDMY_O$ and $HDMY_N$ feature sets are new for the OUH data set, representing an attempt at improving prediction. We use a 5-fold cross-validation with the training data as input and score the models based on the average loss for the fully trained model for each fold. We use early stopping with our validation set and patience of 5 to avoid overfitting during training.

Weight Initialization using Genetic Algorithms For $\mathcal{N}$, we are inspired by Huang et al.'s Poisson neural network (PNN) architecture and implement an ANN with a single hidden layer and gradient descent (GD) as the optimizing function [6]. We create the initial weights for the ANN by using a GA. ANN initialization, specifically line 4 of Figure 2a, is replaced with $\mathcal{N}$ = CreateANNusingGA($\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta}_{\text{ANN}}$) [6]. Using CreateANNusingGA reduces the chance of getting stuck in a local optimum, which is often a problem when using GD. We distinguish between two model architectures that use GA weight initialization, namely MLP_E (exponential activation function) and MLP_L (linear activation function). MLP_E is similar to PNN's use of an exponential activation function for every layer in the architecture [6]. We also study linear activation functions, which capture both negative and positive values, for the output layer in MLP_L when processing a decomposition's residual. We also study Adam as an optimizer in MLP_A [5].

We define chromosomes for CreateANNusingGA as a list containing all the weights of our ANN model. Models making up the GA's initial population are initialized with random weights. In PNN [6], the log maximum likelihood function of a Poisson distribution is used to calculate the fitness of the chromosomes. We replace this with MSE as we do not have a Poisson distribution when working with hourly resolution, as the number of incidents depends on the hour of the day. Furthermore, the residual after decomposition contains negative values, which do not correspond to a Poisson dsitribution. To evaluate the fitness, for

[6] We normalize the input values to the ANN (see Y_N column in Table 1) in order to reduce or avoid issues such as slow convergence, instability, or poor performance when training the ANN. If the data set changes, data preprocessing and normalization before inputting data to the ANN may need to change also.

[7] We use the notation *HDM* for the *Basic* feature set from previous research [5], where it was found to be the best-performing feature set for this data set.

each chromosome we apply the chromosome to an ML model to set weights. We then use this model to make predictions, and calculate fitness using MSE.

CreateANNusingGA uses a two-point crossover to create offspring. Before crossover is applied, the parents are weighted by fitness such that the best-scoring parents have a higher probability of crossover with each other. Mutations are performed using Gaussian mutation, adding a random value with a mean $\mu = 0$ and standard devation $\sigma = 0.1$. We select which chromosomes to keep after each generation, using elitism. Offspring and parents are evaluated equally, and we choose to keep only those with the best fitness. After 50 generations, we set weights using the chromosome with the best fitness in the final evaluation. We then train our model using the entire training data set, optimizing weights with GD. Similarly to the other ANNs, we implement early stopping using the validation set. With DeANN's inclusion of weight initialization using a GA, we aim to improve the previous model [16], while also considering a new domain.

4 Experimental Results

This section discusses experimental results. We use the first three experimental studies—in Section 4.2, Section 4.3, and Section 4.4—to determine the details of the DeANN method including its parameters. Using those results, the forecasting experiments in Section 4.5 demonstrate that the forecasting accuracies of DeANN are better than the accuracies of baseline methods.

4.1 Computing Platform and Baselines

We use Python including the *Keras*[8] library. All baselines are applied to the resulting time series after filtering and preprocessing. Each baseline outputs predictions that can be evaluated using the same metrics used to evaluate our own proposed models. The following baseline models are studied:

- **Simple moving average (SMA)** outputs the average of w previous observations.
- **MEDIC** is implemented using Equation 1.
- **Hermansen and Mengshoel (HM)** uses the best-performing ML model (MLP) and features (HDM) from our earlier research [5].
- **Naive forecast (NF)** uses an observation h steps in the past.

In studies 1, 2, and 3 below we use the validation-split of our data set to perform the model selection. In study 4, in contrast, we evaluate using the test-split to ensure that a complete DeANN model composed from the best-performing components of studies 1, 2, and 3 is not overfitted to the validation set.

Given a forecast horizon h, we are predicting a value at a time $t + h$, where t is the final observation in our training data. We vary h based on our test set and train the model for time steps 0 to t. When we make forecasts that are evaluated, predictions are made in the interval $[t + 1, t + n]$, where n is the size of the test set. In other words, $h \in [1, n]$.

[8] `https://github.com/fchollet/keras`

4.2 Study 1: Feature Selection Results

Goal. We study which features to use for training our models. The $\boldsymbol{x} = HDM$ features gave the best results earlier [5], now we consider adding year Y (see Table 1). We want to see if including year, either as a numeric value in $\boldsymbol{x} = HDMY_N$ or via one-hot encoding in $\boldsymbol{x} = HDMY_O$, will improve forecasts.

Method and Data. To evaluate the effect of the features and keep the results consistent, we use an existing ANN architecture [5] but adjust for the new features. The architecture consists of two hidden layers with 16 nodes in each layer, using the *Adam* optimizer. This study and Study 3 are performed with random windows of 10% of our full data set, to maintain a time series, then evaluated via average scores over five iterations. This results in a higher variation of scores between the experiments but counteracts possible overfitting on the full data set.

Results. From Table 2, we see that including the year as a numeric value improves our model performance. We consequently favor the use of year Y encoded as a numeric value, $\boldsymbol{x} = HDMY_N$, rather than $\boldsymbol{x} = HDMY_O$.

Table 2: Validation error for the same ANN architecture, but with different input parameters without decomposition. *HDM* stands for hour, day of the week, and month. MSE is an average.

Features $\boldsymbol{x}$	**MSE**
HDM	22.63
$HDMY_N$	**21.73**
$HDMY_O$	24.60

4.3 Study 2: Statistical Decomposition using STL and SSA

Goal. Our main objective with decomposition using STL and SSA is to extract as many patterns as possible from our time series, leaving less complexity to handle for ML. Which decomposition method works best for our data?

Method and Data. As we have not found literature supporting the choice of one decomposition method over others, we test possible combinations and evaluate them. The following statistical decomposition implementations are used for the methods discussed in Section 3.2:

- **STL:** The *statsmodels*-library for Python [11] performs STL-decomposition. We create predictions by summing the extended seasonality and predicted trend obtained from polynomial regression as described for STL in Section 3.2.
- **SSA:** We use *RSSA* [3], an SSA implementation in R. We use RSSA's built-in forecasting methods to compute forecasts from a decomposition: *bootstrap*, *recurrent forecasting*, and *vector* [3]. The remaining code is in Python, thus *RPy2*[9] provides an interface to run embedded R in Python.

We first find the parameters for decomposition best suited for our problem for SSA. Then, we find the best approximation with polynomial regression based on the trend from STL.

[9] `https://rpy2.github.io/doc/latest/html/index.html`

Table 3: SSA parameter selection results.

Grouping	**Forecasting**	**MSE**	**Grouping**	**Forecasting**	**MSE**
W-correlation	Bootstrap	25.33	Periodgram	Bootstrap	25.50
W-correlation	Vector	26.30	Periodgram	Vector	26.58
W-correlation	Recurrent	**24.60**	Periodgram	Recurrent	25.32

Results. The results from Table 3 show that SSA with W-correlation matrix and recurrent forecasting gave the best results for SSA-decomposition. As for STL, results from Table 4 suggest that using a weekly period for seasonality generally scores better than a daily period. The results concur with our data analysis, suggesting that the ambulance demand pattern varies across days of the week. Additionally, a second-degree polynomial minimizes error on our validation data.

By graphing polynomials of varying degrees n, we observe a negative gradient for $n = 2$ and $n = 5$ towards the end of our data, as well as for the trend with an interpolated period. Discussions with OUH and our analysis suggest, in contrast, that the trend is increasing annually.[10] Thus, we assume the best-scoring STL-decomposition with an increasing trend towards the end of our time series will perform well.

The discussion above and Table 4 suggest the following. First, STL is preferred to SSA. Second, with an MSE of 23.75, we keep using weekly seasonality $p_s = 7$ and third-degree polynomial regression (STL$_3$) for predicting the trend. We also keep the second-degree polynomial regression model (STL$_2$) with weekly seasonality $p_s = 7$ for our final study, as it has the best MSE of 23.59.

Table 4: STL and polynomial regression parameter selection results: varying degree $n \in \{1, 2, 3, 4, 5\}$ and daily ($p_s = 1$) or weekly ($p_s = 7$) period for STL's seasonal component. The trend component to approximate is based on a $p_t = 365$ day period.

n	p_s	**MSE**	n	p_s	**MSE**
1	1	25.66	1	7	24.17
2	1	25.08	2	7	**23.59**
3	1	25.24	3	7	23.75
4	1	25.93	4	7	24.44
5	1	25.12	5	7	23.63

4.4 Study 3: Weight Initialization with GA

Goal. The experiment aims to study ANN weight initialization with GAs for model improvement. Additionally, we want to see how using a linear activation function for the output layer measures up against exponential activation.

Methods and Data. We evaluate the performance of our MLP neural network after setting initial weights using CreateANNusingGA, doing no backpropagation to update weights further, to evaluate the initial performance of our model.

[10] This increase reflects the time period covered by the data set, the years 2015–2019. It was feasible to extract data for this period from the EMS system of OUH, given the resources available.

Table 5: Weight initialization with GA results.

Activation	**Initialization**	**MSE**	**Activation**	**Initialization**	**MSE**
Exponential	Random	21.98	Linear	Random	21.88
Exponential	GA	**21.23**	Linear	GA	21.54

Results. The results from this experiment are presented in Table 5. Both models show improvement when CreateANNusingGA initializes weights before training. The results indicate that exponential activation for the final layer yields better forecasts. Based on these results, we will use GAs for weight initialization for this model. We will continue to use exponential activation when data is not decomposed. We will refer to the architecture as MLP_L when a linear activation function is used and MLP_E when an exponential activation function is used.

4.5 Study 4: Final Forecasting Experiments

Goal. This section discusses our final forecasting results. We present how variants of the DeANN method compare to each other and to baselines. The study includes discussions of how decomposition can contribute to forecasting.
Method and Data. To evaluate the final predictions, forecasting performance metrics MAE and MSE for DeANN variants are compared to those of several baselines as identified in Section 4.1. After experiments with different values for p_s and p_t with STL (see Section 4.3), we use $p_s = 7$ and $p_t = 365$ in the final forecasting experiments.
Results. Table 6 presents results for different variants of our method. Using MSE, the DeANN method with the highest forecasting accuracy is $\mathcal{M}_{15}$. This is the PNN-inspired architecture, MLP_L, with a linear activation function, features $\boldsymbol{x} = \mathit{HDMY}_N$, and STL with a third-degree polynomial regression to approximate the trend. Model $\mathcal{M}_{10}$ with MLP_A (the MLP architecture with *Adam* optimizer) and using STL decomposition with a second-degree polynomial to estimate the trend is the model that gave the best predictions using MAE.

Table 7 presents forecasting results of the DeANN variant $\mathcal{M}_{15}$ relative to baselines from the literature.[11] The main point here is that our novel $\mathcal{M}_{15}$ model outperforms the baselines. While the improvement is relatively small, it can result in more accurate decision-making in the complex and high-stakes area of EMS decision-making, which is very valuable. Figure 3 contains forecasts from our best-performing DeANN variants $\mathcal{M}_{10}$ and $\mathcal{M}_{15}$ in terms of MSE and MAE on a week from our test set. The models follow each other closely, with the MAE model generally predicting a lower demand than the MSE model. The actual demand includes a lot of stochasticity, which is what we attempt to predict with our residual-predicting ANN. Due to the data's stochastic and complex nature [12,5,10], it is impossible to forecast the actual demand perfectly. However, the overall trend and seasonality of the actual demand are reasonably captured by

[11] The statistical decomposition is only used with DeANN, as presented in Figure 2, not with the baselines. MEDIC, for example, is applied on the raw data. Future research could integrate decomposition and MEDIC, for example.

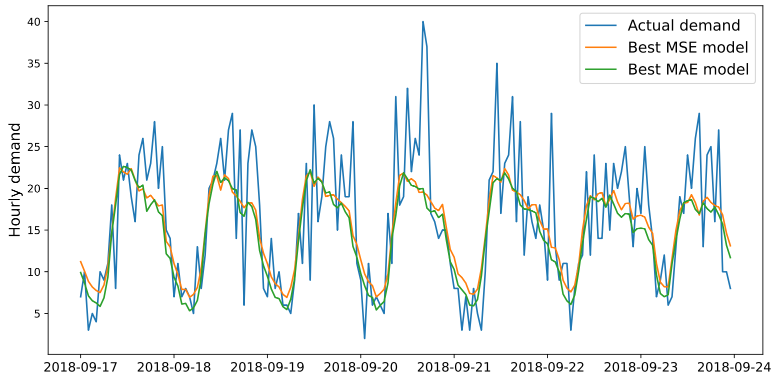

Fig. 3: Forecasts of our best models (see Table 6) using MSE, $\mathcal{M}_{15}$, and MAE, $\mathcal{M}_{10}$, for a week in September, compared to the actual demand that week.

Table 6: Comparison of different variants of our method, with and without decomposition (De). The best results are in bold; our $\mathcal{M}_{15}$ method has the best MSE. State of the art on this data set prior to $\mathcal{M}_{15}$ is $\mathcal{M}_1$ [5].

Id	**De**	**Model**	**MSE**	**MAE**	**Id**	**De**	**Model**	**MSE**	**MAE**
$\mathcal{M}_1$	None	MLP_A, HDM	22.95	3.69	$\mathcal{M}_9$	None	MLP_A, $HDMY_N$	21.77	3.62
$\mathcal{M}_2$	STL$_2$	MLP_A, HDM	22.03	3.63	$\mathcal{M}_{10}$	STL$_2$	MLP_A, $HDMY_N$	21.77	**3.61**
$\mathcal{M}_3$	STL$_3$	MLP_A, HDM	21.74	3.64	$\mathcal{M}_{11}$	STL$_3$	MLP_A, $HDMY_N$	21.73	3.64
$\mathcal{M}_4$	SSA	MLP_A, HDM	21.93	3.65	$\mathcal{M}_{12}$	SSA	MLP_A, $HDMY_N$	21.92	3.64
$\mathcal{M}_5$	None	MLP_E, HDM	25.39	3.85	$\mathcal{M}_{13}$	None	MLP_E, $HDMY_N$	21.95	3.66
$\mathcal{M}_6$	STL$_2$	MLP_L, HDM	21.94	3.63	$\mathcal{M}_{14}$	STL$_2$	MLP_L, $HDMY_N$	21.92	3.62
$\mathcal{M}_7$	STL$_3$	MLP_L, HDM	21.74	3.65	$\mathcal{M}_{15}$	STL$_3$	MLP_L, $HDMY_N$	**21.68**	3.64
$\mathcal{M}_8$	SSA	MLP_L, HDM	21.85	3.64	$\mathcal{M}_{16}$	SSA	MLP_L, $HDMY_N$	21.98	3.65

our model, along with some hard-to-decompose structure captured by the ANN. Unfortunately, the high peaks and deep valleys in incident data are extremely difficult to forecast.

One key takeaway from these results is the importance of capturing the trend in our time series. We deduce the importance of the trend by examining the only two models where the $\boldsymbol{x} = HDM$ features were used with no decomposition. These two models, $\mathcal{M}_1$ and $\mathcal{M}_5$ in Table 6, gave the worst results.

Comparing results using decomposition and $\boldsymbol{x} = HDMY_N$ features with results using non-decomposed data further underlines the importance of the decomposition and including year as a feature. The results from decomposition indicate that using STL improves the forecasts of our MLP-models and outperforms decomposition using SSA. SSA improved the accuracy of forecasts from MLP when year was not included in the input data, but produced similar results as not using decomposition at all when year was included.

5 Conclusion and Future Work

We have developed a neuro-symbolic ML approach, DeANN, for ambulance demand forecasting. DeANN integrates statistical decomposition and ANNs. Our experimental forecasting results suggest that our proposed method outperform existing baselines. We improve on existing methods by improving input features, the architecture of the model, and using decomposition to pre-process data before inputting them to an MLP. We found success in using STL decomposition when including the year as a numeric value among input features. Our results indicate that decomposition with STL and SSA can improve model accuracy while also being understandable. STL has somewhat better results than SSA in our experiments. Our intuition that extracting seasonal and trend components of the time series will simplify the problem for the ML model to learn has been confirmed by these results.

Although we made some improvements by using decomposition and ML, perhaps the approach could be further studied and expanded: by considering decomposition using more finely tuned parameters; by using other (including online) ML methods, both neural and symbolic;[12] and by integrating with other emergency management services including ambulance allocation [9]. Ultimately, one could perhaps use improved forecasting results to better perform (i) ambulance and human resource management and (ii) placement of ambulances if the spatial dimension is introduced into the forecasts.

Table 7: Comparison of the best method $\mathcal{M}_{15}$ in Table 6 (bottom row) and four baselines (top four rows; $\mathcal{M}_1$ in fourth). The best results, for $\mathcal{M}_{15}$, are in bold.

De	Model	MSE	MAE
-	SMA, $w = 6$ hrs	46.12	5.45
-	NF, $h = 7$ days	42.37	5.02
-	MEDIC	26.33	3.91
-	MLP_A, HDM	22.95	3.69
STL$_3$	MLP_L, $HDMY_N$	**21.68**	**3.64**

Acknowledgements

We would like to acknowledge Oslo University Hospital, and in particular Jo Kramer-Johansen, for providing the EMS incident data set, and for their shared enthusiasm and collaboration on this project. Helpful comments from anonymous reviewers are also acknowledged.

References

1. Basak, A., Mengshoel, O.J., Kulkarni, C., Schmidt, K., Shastry, P., Rapeta, R.: Optimizing the decomposition of time series using evolutionary algorithms: soil moisture analytics. In: Proc. GECCO. pp. 1073–1080 (2017)

[12] For example, the Deep Learning library for Time Series and Sequences may be studied: `https://github.com/timeseriesAI/tsai`.

2. Cleveland, R.B., Cleveland, W.S., McRae, J.E., Terpenning, I.: STL: A seasonal-trend decomposition procedure based on Loess. Journal of Official Statistics **6**(1), 3–73 (1990)
3. Golyandina, N., Korobeynikov, A.: Basic singular spectrum analysis and forecasting with R. Computational Statistics & Data Analysis **71**, 934–954 (2014)
4. Golyandina, N., Korobeynikov, A., Zhigljavsky, A.: Singular Spectrum Analysis with R (2010)
5. Haugsbø Hermansen, A., Mengshoel, O.J.: Forecasting ambulance demand using machine learning: A case study from Oslo, Norway. In: 2021 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 1–10 (2021)
6. Huang, H., Jiang, M., Ding, Z., Zhou, M.: Forecasting emergency calls with a Poisson neural network-based assemble model. IEEE Access **7**, 18061–18069 (2019)
7. Lara-Benítez, P., Carranza-García, M., Riquelme, J.C.: An experimental review on deep learning architectures for time series forecasting. International Journal of Neural Systems **31**(3) (March 2021)
8. Makridakis, S., Spiliotis, E., Assimakopoulos, V.: Statistical and machine learning forecasting methods: Concerns and ways forward. PLOS ONE **13**(3), 1–26 (2018)
9. Mukhopadhyay, A., Pettet, G., Vazirizade, S.M., Lu, D., Jaimes, A., El Said, S., Baroud, H., Vorobeychik, Y., Kochenderfer, M., Dubey, A.: A review of incident prediction, resource allocation, and dispatch models for emergency management. Accident Analysis & Prevention **165**, 106501 (2022)
10. Schjølberg, M.E., Bekkevold, N.P., Sánchez-Díaz, X., Mengshoel, O.J.: Comparing metaheuristic optimization algorithms for ambulance allocation: An experimental simulation study. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 1454—1463 (2023)
11. Seabold, S., Perktold, J.: statsmodels: Econometric and statistical modeling with Python. In: 9th Python in Science Conference (2010)
12. Setzler, H., Saydam, C., Park, S.: EMS call volume predictions: A comparative study. Computers & Operations Research **36**, 1843–1851 (2009)
13. Soeffker, N., Ulmer, M.W., Mattfeld, D.C.: Stochastic dynamic vehicle routing in the light of prescriptive analytics: A review. European Journal of Operational Research **298**(3), 801–820 (2022)
14. Van De Weijer, E., Owren, O.A.: Forecasting Ambulance Demand in Oslo and Akershus. Master's thesis, Norwegian University of Science and Technology (NTNU) (2022), `https://hdl.handle.net/11250/3030248`
15. Wu, C.L., Chau, K.W., Fan, C.: Prediction of rainfall time series using modular artificial neural networks coupled with data-preprocessing techniques. Journal of Hydrology **389**(1), 146–167 (2010)
16. Zhang, Y., Li, G., Muskat, B., Law, R.: Tourism demand forecasting: A decomposed deep learning approach. Journal of Travel Research **60**(5), 981–997 (2021)
17. Zhou, Z., Matteson, D.S.: Predicting ambulance demand: a spatio-temporal kernel approach. In: Proc. KDD. p. 2297–2303 (2015)